

# Theoretical Framework for Comparing Several Stochastic Optimization Approaches

James C. Spall, Stacy D. Hill and David R. Stark

The Johns Hopkins University  
Applied Physics Laboratory  
11100 Johns Hopkins Road  
Laurel, Maryland 20723-6099 U.S.A.  
E-mail: james.spall@jhuapl.edu

## Abstract

This paper establishes a framework for formal comparisons of several leading optimization algorithms, establishing guidance to practitioners for when to use or not use a particular method. The focus in this paper is five general algorithm forms: random search, simultaneous perturbation stochastic approximation, simulated annealing, evolutionary strategies, and genetic algorithms. We summarize the available theoretical results on rates of convergence for the five algorithm forms and then use the theoretical results to draw some preliminary conclusions on the relative efficiency. Our aim is to sort out some of the competing claims of efficiency and to suggest a structure for comparison that is more general and transferable than the usual problem-specific numerical studies.

**Keywords:** Stochastic optimization; rate of convergence; random search; simultaneous perturbation stochastic approximation (SPSA); simulated annealing; evolutionary computation; genetic algorithms.

## 1. INTRODUCTION

To address the shortcomings of classical deterministic algorithms, a number of powerful optimization algorithms with embedded randomness have been developed. The population-based methods of evolutionary computation are only one class among many of these available *stochastic* optimization algorithms. Hence, a user facing a challenging optimization problem for which a stochastic optimization method is appropriate meets the daunting task of determining which algorithm is appropriate for a given problem. This choice is made more difficult by some dubious claims that have been made about some popular algorithms. An inappropriate approach may lead to a large waste of resources, both from the view of wasted efforts in

implementation and from the view of the resulting suboptimal solution to the optimization problem of interest.

Hence, there is a need for objective analysis of the relative merits and shortcomings of leading approaches to stochastic optimization. This need has certainly been recognized by others, as illustrated in recent conferences on evolutionary computation, where numerous sessions are devoted to comparing algorithms. Nevertheless, virtually all comparisons have been numerical tests on specific problems. Although sometimes enlightening, such comparisons are severely limited in the *general* insight they provide. Some comparisons for *noisy* evaluations of a simple spherical loss function are given in Arnold (2002, Chap. 6); however, some of the competitors were implemented in non-standard forms, making the results difficult to interpret for an analyst using a more conventional implementation. Spall (2003) also has a number of comparisons (theoretical and numerical) for the cases of noise-free and noisy loss evaluations. On the other end of the spectrum are the “No Free Lunch Theorems” (Wolpert and McReady, 1997), which simultaneously consider all possible loss functions and thereby draw conclusions that have limited practical utility since one always has at least *some* knowledge of the nature of the loss function being minimized.

Our aim in this paper is to lay a framework for a *theoretical* comparison of efficiency applicable to a broad class of practical problems where some (incomplete) knowledge is available about the nature of the loss function. We will consider five basic algorithm forms—random search, simultaneous perturbation stochastic approximation (SPSA), simulated annealing (SAN), and two forms of evolutionary computation (evolution strategy and genetic algorithms). The basic optimization problem corresponds to finding an optimal point  $\theta^*$ :

$$\theta^* = \arg \min_{\theta \in \Theta} L(\theta),$$

---

This work was partially supported by the JHU/APL IRAD Program and U.S. Navy Contract N00024-98-D-8124. An expanded version of this paper is available upon request.

where  $L(\theta)$  is the loss function to be minimized,  $\Theta$  is the domain over which the search will occur, and  $\theta$  is a  $p$ -dimensional (say) vector of parameters. We are mainly interested in the case where  $\theta^*$  is a *unique* global minimum.

Although stochastic optimization approaches other than the five above exist, we are restricting ourselves to the five general forms in order to be able to make tangible progress (note that there are various specific implementations of each of these general algorithm forms). These five algorithms are general-purpose optimizers with powerful capabilities for serious multivariate optimization problems. Further, they have in common the requirement that they only need measurements of the objective function, not requiring derivative information (gradient or Hessian) for the loss function.

One might ask whether questions of relative efficiency are relevant in light of the “no free lunch (NFL)” theorems of Wolpert and Macready (1997) and others. The NFL theorems state, in essence, that the expected performance of any pair of optimization algorithms across all possible problems is identical. In practice, of course, one is not interested in solving “all possible problems,” as there is usually some prior information about the problems of interest and this prior information will affect the algorithm implementation. Hence, the NFL results may not adequately reflect the performance of candidate algorithms as they are actually applied. In other words, some algorithms *do* work better than others on problems of interest. Nevertheless, the NFL results are an important backdrop against which to view the results here, providing limits on the extent to which one algorithm can be claimed as “better” than another.

## 2. SIMPLE GLOBAL RANDOM SEARCH

We first establish a rate of convergence result for the simplest random search method where we repeatedly sample over the domain of interest,  $\Theta \subseteq \mathbb{R}^p$ . This can be done in recursive form or in “batch” (non-recursive) form by simply laying down a number of points in  $\Theta$  and taking as our estimate of  $\theta^*$  that value of  $\theta$  yielding the lowest  $L$  value.

To evaluate the rate, let us specify a “satisfactory region”  $S(\theta^*)$  representing some neighborhood of  $\theta^*$  providing acceptable accuracy in our solution (e.g.,  $S(\theta^*)$  might represent a hypercube about  $\theta^*$  with the length of each side representing a tolerable error in each coordinate of  $\theta$ ). An expression related to the rate of convergence of the above simple random search algorithm is then given by

$$P(\hat{\theta}_k \in S(\theta^*)) = 1 - [1 - P(\theta_{\text{new}}(k) \in S(\theta^*))]^k \quad (2.1)$$

We will use this expression in Section 7 to derive a convenient formula for comparison of efficiency with other algorithms.

## 3. SIMULTANEOUS PERTURBATION STOCHASTIC APPROXIMATION

The next algorithm we consider is SPSA. This algorithm is designed for continuous variable optimization problems. Unlike the other algorithms here, SPSA is fundamentally oriented to the case of *noisy* function measurements and most of the theory is in that framework. This will make for a difficult comparison with the other algorithms, but Section 7 will attempt a comparison nonetheless. The SPSA algorithm works by iterating from an initial guess of the optimal  $\theta$ , where the iteration process depends on a highly efficient “simultaneous perturbation” approximation to the gradient  $g(\theta) \equiv \partial L(\theta)/\partial \theta$ .

Assume that measurements  $y(\theta)$  of the loss function are available at any value of  $\theta$ :

$$y(\theta) = L(\theta) + \text{noise}.$$

For example, in a Monte Carlo simulation-based optimization context,  $L(\theta)$  may represent the mean response with input parameters  $\theta$ , and  $y(\theta)$  may represent the outcome of one simulation experiment at  $\theta$ . In some problems, exact loss function measurements will be available; this corresponds to the *noise* = 0 setting (and in the simulation example, would correspond to a deterministic—non-Monte Carlo—simulation). Note that no direct measurements (with or without noise) of the gradient of  $L(\theta)$  are assumed available.

The SPSA procedure is in the general recursive SA form:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \hat{g}_k(\hat{\theta}_k), \quad (3.1)$$

where  $\hat{g}_k(\hat{\theta}_k)$  is the estimate of the gradient  $g(\theta)$  at the iterate  $\hat{\theta}_k$  based on the above-mentioned measurements of the loss function and  $a_k > 0$  is a “gain” sequence. This iterate can be shown to converge under reasonable conditions (e.g., Spall, 1992, and Dippon and Renz, 1997, for local convergence; Maryak and Chin, 2001, for global convergence). The essential basis for efficiency of SPSA in multivariate problems is due to the gradient approximation, where only two measurements of the loss function are needed to estimate the  $p$ -dimensional gradient vector for any  $p$ ; this contrasts with the standard finite difference method of gradient approximation, which requires  $2p$  measurements.

Most relevant to the comparative analysis goals of this paper is the asymptotic distribution of the iterate. This was derived in Spall (1992), with further developments in Chin (1997), Dippon and Renz (1997), and Spall (2000). Essentially, it is known that under appropriate conditions,

$$k^{\beta/2}(\hat{\theta}_k - \theta^*) \xrightarrow{\text{dist}} N(\mu, \Sigma) \text{ as } k \rightarrow \infty, \quad (3.2)$$

where  $\beta > 0$  depends on the choice of gain sequences ( $a_k$  and  $c_k$ ),  $\mu$  depends on both the Hessian and the third derivatives of  $L(\theta)$  at  $\theta^*$  (note that in general,  $\mu \neq 0$  in contrast to many well-known asymptotic normality results in estimation), and  $\Sigma$  depends on the Hessian matrix at  $\theta^*$  and the variance of the noise in the loss measurements. Given the restrictions on the gain sequences to ensure convergence and asymptotic normality, the fastest allowable value for the rate of convergence of  $\hat{\theta}_k$  to  $\theta^*$  is  $k^{-1/3}$ . This contrasts with the fastest allowable rate of  $k^{-1/2}$  for gradient-based algorithms such as Robbins-Monro SA.

Unfortunately, (3.2) is not directly usable in our comparative studies here since the other three algorithms being considered here appear to have convergence rate results only for the case of *noise-free* loss measurements. The authors are unaware of any general asymptotic distribution result for the noise-free case (note that it is *not* appropriate to simply let the noise level go to zero in (3.2) in deriving a result for the noise-free case; it is likely that the rate factor  $\beta$  will also change if an asymptotic distribution exists). Some partial results, however, are available that are related to the rate of convergence. Gerencsér (1999) established that the moments

$\left[ E \left( \left\| \hat{\theta}_k - \theta^* \right\|^q \right) \right]^{1/q}$  converge to zero at a rate of  $k^{-1/2}$  for

any  $q > 0$ , when  $a_k$  has the standard  $1/k$  decay rate. More recently, Gerencsér and Vágó (2000) established that the noise-free SPSA algorithm has a geometric rate of convergence when *constant* gains  $a_k = a$  are used. In particular, for functions having bounded third derivatives, they show for sufficiently small  $a$ ,

$$\limsup_{k \rightarrow \infty} \frac{\left\| \hat{\theta}_k - \theta^* \right\|}{\eta^k} = 1 \text{ a.s.}$$

for some  $0 < \eta < 1$ . Gerencsér and Vágó (2000) go further for quadratic loss functions by specifying  $\eta$  in terms of  $a$  and the Hessian matrix of  $L$ . Unfortunately, even in the quadratic case,  $\eta$  is not fully specified in terms of quantities associated with  $L$  and the algorithm itself (i.e.,  $\eta$  depends on unknown constants).

#### 4. SIMULATED ANNEALING ALGORITHMS

The SAN method (Metropolis et al., 1953; Kirkpatrick et al., 1983) was originally developed for optimization over discrete finite sets. The Metropolis SAN method produces a sequence that converges in probability to the set of global minima of the loss function as  $T_k$ , the *temperature*, converges to zero.

Gelfand and Mitter (1993) present a SAN method for continuous parameter optimization. They obtained discrete-time recursions (which are similar to a stochastic approximation algorithm) for Metropolis-type SAN

algorithms that, in the limit, optimize continuous parameter loss functions.

Furthermore, like SPSA, SAN has an asymptotic normality result (but unlike SPSA, this result applies in the noise-free case). Let  $H(\theta^*)$  denote the Hessian of  $L(\theta)$  evaluated at  $\theta^*$  and let  $I_p$  denote the  $p \times p$  identity matrix. Yin (1999) showed that for  $b_k = (b/(k^\gamma \log(k^{1-\gamma} + B_0)))^{1/2}$ ,

$$[\log(k^{1-\gamma} + B_0)]^{1/2}(\hat{\theta}_k - \theta^*) \rightarrow N(0, \Sigma) \text{ in distribution,}$$

where  $\Sigma H + H^T \Sigma + (b/a)I = 0$ .

#### 5. EVOLUTIONARY COMPUTATION: EVOLUTIONARY STRATEGIES

There are three general approaches in evolutionary computation (EC), namely Evolutionary Programming (EP), Evolutionary Strategies (ES) and Genetic Algorithms (GA). All three approaches work with a population of candidate solutions and randomly alter the solutions over a sequence of generations according to evolutionary operations of competitive selection, mutation and sometimes recombination (reproduction). The fitness of each population element to survive into the next generation is determined by a selection scheme based on evaluating the loss function for each element of the population. The selection scheme is such that the most favorable elements of the population tend to survive into the next generation while the unfavorable elements tend to perish.

The principle differences in the three approaches are the selection of evolutionary operators used to perform the search and the computer representation of the candidate solutions. EP uses selection and mutation only to generate new solutions. While both ES and GA use selection, recombination and mutation, recombination is used more extensively in GA. A GA traditionally performs evolutionary operations using binary encoding of the solution space, while EP and ES perform the operations using real-coded solutions. The GA also has a real-coded form and there is some indication that the real-coded GA may be more efficient and provide greater precision than the binary-coded GA. The distinction among the three approaches has begun to blur as new hybrid versions of EC algorithms have arisen.

Global convergence results can be given for a broad class of problems, but the same cannot be said for convergence *rates*. Both Beyer (1995) and Rudolph (1997a) examine ES algorithms that include selection, mutation and recombination. The function analyzed in both cases is the classic spherical fitness function  $L(\theta) = \|\theta\|^2$  whose exact solution is of course known. Convergence rates based on the spherical fitness function are somewhat useful, if it is assumed that the sphere approximates a local basin of attraction. A number of other convergence rate results are also available for that fitness function, for example Qi and Palmeiri (1994) for real-valued GA. The most practically useful convergence rates for EC algorithms seem to be for the class of strongly convex fitness

functions. The following theorem due to Rudolph (1997b) is an application of a more general result by Rappel (1989). The theorem will be the starting place for the specific convergence rate result that will be used for comparison in Section 7.

An EC algorithm has a *geometric rate of convergence* if and only if  $E[L_k^* - L(\theta^*)] = O(\eta^k)$  where  $\eta \in (0, 1)$  is called the *convergence rate*. Under conditions, the convergence rate result for a  $(1, \lambda)$ -ES using only selection and mutation on a  $(K, Q)$ -strongly convex fitness function is geometric with a rate of convergence

$$\eta = (1 - M_{\lambda,p}^2 Q^2)$$

where  $Q$  is a constant,  $M_{\lambda,p} = E[B_{\lambda,\lambda}] > 0$ , and where  $B_{\lambda,\lambda}$  denotes the maximum of  $\lambda$  independent identically distributed Beta random variables. The computation of  $M_{\lambda,p}$  is complicated since it depends on both the number of offspring  $\lambda$  and the problem dimension  $p$ . Asymptotic approximations are available and will be shown next. Assuming  $p$  is fixed and  $\lambda \rightarrow \infty$  then  $M_{\lambda,p} \approx (2 p^{-1} \log \lambda)^{1/2}$ . To extend this convergence rate from a  $(1, \lambda)$ -ES to a  $(N_{\text{pop}}, \lambda)$ -ES, note that each of the  $N_{\text{pop}}$  parents generate  $\lambda/N_{\text{pop}}$  offspring. Then the convergence rate for the  $(N_{\text{pop}}, \lambda)$ -ES where offspring are only obtained by mutation is

$$\eta \leq [1 - (2p^{-1} \log(\lambda/N_{\text{pop}}))/Q^2]$$

for  $(K, Q)$ -strongly convex functions.

## 6. EVOLUTIONARY COMPUTATION: GENETIC ALGORITHMS

As discussed in Stark and Spall (2001), it is possible to cast the GA in the framework of Markov chains. This allows for a rate of convergence analysis. Consider a GA with a population size of  $N$ . Further, suppose that each population element is a binary string of length  $b$  bits. Hence, there are  $2^b$  possible strings for an *individual* population element. Then the total number of possible populations is given by

$$N_{\text{pop}} \equiv \frac{(N + 2^b - 1)!}{(2^b - 1)! N!}.$$

It is possible to construct a Markov transition matrix  $\Pi$  that provides the probability of transitioning from one population of size  $N$  to another population of the same size. This transition matrix is  $N_{\text{pop}} \times N_{\text{pop}}$ . An individual element in the transition matrix can be computed according to the formulas in Stark and Spall (2001) (see also Suzuki, 1995). These elements depend in a non-trivial way on the population size, crossover rate, mutation rate, and number of elements considered “elite.”

Of primary interest in analyzing the performance of GA algorithms using Markov chains is the probability of obtaining a population that contains the optimum  $\theta^*$ . Let  $\pi_k$  be an  $N \times 1$  vector having  $j^{\text{th}}$  component,  $\pi_k(j)$ , equal to the probability that the  $k^{\text{th}}$  generation will result in population  $j$ . From basic Markov chain theory,

$$\pi_k^T = \pi_{k-1}^T \Pi = \pi_0^T \Pi^k$$

where  $\pi_0$  is an initial probability distribution.

The stationary distribution of the GA is then given by

$$\bar{\pi}^T \equiv \lim_{k \rightarrow \infty} \pi_k^T = \lim_{k \rightarrow \infty} \pi_0^T \Pi^k.$$

Further, under standard ergodicity assumptions for Markov chains,  $\bar{\pi}$  satisfies  $\bar{\pi}^T = \bar{\pi}^T \Pi$ . This equation provides a mechanism for solving directly for the stationary distribution (e.g., Iosifescu, 1980, pp. 123–124).

Unfortunately, from a practical view, the Markov chain approach has a significant deficiency. The dimension  $N$  grows very rapidly with increases in the number of bits  $b$  and/or the population size  $N$ . A perhaps more intuitive estimate of the size of  $N_{\text{pop}}$  can be obtained by Stirling’s Approximation as follows:

$$N_{\text{pop}} \approx \sqrt{2\pi} \left(1 + \frac{2^b - 1}{N}\right)^N \left(1 + \frac{N}{2^b - 1}\right)^{2^b - 1} \left(\frac{1}{2^b - 1} + \frac{1}{N}\right)^{1/2}$$

Thus far, our analysis using the above approach has been restricted to scalar  $\theta$  systems (requiring fewer bits  $b$  than a multivariate system) and low  $N_{\text{pop}}$ .

## 7. COMPARATIVE ANALYSIS

### 7.1 Problem Statement and Summary of Efficiency Theory for the Five Algorithms

This section uses the specific algorithm results in Sections 2 to 6 above in drawing conclusions on the relative performance of the five algorithms. There are obviously many ways one can express the rate of convergence, but it is expected that, to the extent they are based on the theory outlined above, the various ways will lead to broadly similar conclusions. We will address the rate of convergence by focusing on the question:

*With some high probability  $1 - \rho$  ( $\rho$  a small number), how many  $L(\cdot)$  function evaluations, say  $n$ , are needed to achieve a solution lying in some “satisfactory set”  $S(\theta^*)$  containing  $\theta^*$ ?*

With the random search algorithm in Section 2, we have a closed form solution for use in questions of this sort while with the SPSA, SAN, and EC algorithms of Sections 3 through 5, we must apply the existing asymptotic results, assuming that they apply to the finite-sample question above. For the GA, there is a finite sample solution using the Markov chain approach. For each of the five algorithms, we will outline below an analytical expression

useful in addressing the question. After we have discussed the analytical expressions, we present a comparative analysis in a simple problem setting for varying  $p$ .

### Random Search

We can use (2.1) to answer the question above. Setting the left-hand side of (2.1) to  $1 - \rho$  and supposing that there is a constant sampling probability  $P^* = P(\theta_{\text{new}}(k) \in S(\theta^*)) \forall k$ , we have

$$n = \frac{\log \rho}{\log(1 - P^*)}. \quad (7.1)$$

### Simultaneous Perturbation Stochastic Approximation

From the fact that SPSA uses two  $L(\theta^*)$  evaluations per iteration, the value  $n$  to achieve the desired probability for  $\hat{\theta}_k \in S(\theta^*)$  is then

$$n = 2 \left( \frac{2d(p)\sigma}{\delta s} \right)^3$$

where from standard  $N(0, 1)$  distribution tables, there exists a displacement factor, say  $d(p)$ , such that the probability contained within  $\pm d(p)$  units contains probability amount  $(1 - \rho)^{1/p}$ . We are interested in the  $k$  such that  $2d(p)\sigma/k^{1/3} = \delta s = s_i^+ - s_i^-$  (the common length of a side in a  $p$ -fold hypercube).

### Simulated Annealing

The value  $n$  to achieve the desired probability for  $\hat{\theta}_k \in S(\theta^*)$  is

$$\log n = \frac{1}{1 - \gamma} \left( \frac{2d(p)\sigma}{\delta s} \right)^2.$$

### Evolutionary Strategy

As discussed in Section 6, the rate-of-convergence results for algorithms of the evolutionary computation type are not as well developed as for the other three algorithms of this paper. Theorem 6.1 gives a general bound on  $E[L(\hat{\theta}_k) - L(\theta^*)]$  for application of a  $(N, \lambda)$ -ES form of EC algorithm to strongly convex functions. A more explicit form of the bound is available for the  $(1, \lambda)$ -ES. Unfortunately, even in the optimistic case of an explicit numerical bound on  $E[L(\hat{\theta}_k) - L(\theta^*)]$ , we cannot readily translate the bound into a probability calculation for  $\hat{\theta}_k \in S(\theta^*)$ , as used above (and, conversely, the asymptotic normality result on  $\hat{\theta}_k$  for SPSA and SAN cannot be readily translated into one on  $L(\hat{\theta}_k)$  since  $\partial L / \partial \theta = 0$  at  $\theta^*$ —see, e.g., Serfling, 1980, pp. 122–124—although

Lehmann, 1983, pp. 338–339 suggests a possible means of coping with this problem via higher-order expansions). So, in order to make some reasonable comparison, let us suppose that we can associate a set  $S(\theta^*)$  with a given deviation from  $L(\theta^*)$ , i.e.,  $S(\theta^*) = S(\theta^*, \varepsilon) = \{\theta: L(\hat{\theta}_k) - L(\theta^*) \leq \varepsilon\}$  for some prespecified tolerance  $\varepsilon > 0$ . As presented in Rudolph (1997b),  $E[L(\hat{\theta}_k) - L(\theta^*)] \leq c^k$  for sufficiently large  $k$ , where  $c$  is the convergence rate in Section 6. Then by Markov's inequality,

$$1 - P(\hat{\theta}_k \in S(\theta^*)) \leq \frac{E[L(\hat{\theta}_k) - L(\theta^*)]}{\varepsilon} \leq \frac{c^k}{\varepsilon}, \quad (7.2)$$

indicating that  $P(\hat{\theta}_k \in S(\theta^*))$  is bounded below by the ES bounds mentioned in Section 5.

The full version of the paper employs Markov's inequality and the bound in Rudolph (1977b) to show that there are  $\lambda$  evaluations of the fitness function for each generation  $k$  so that  $n = \lambda k$ , where

$$k = \frac{\log \rho - \log(1/\varepsilon)}{\log \left[ 1 - \frac{2}{pQ^2} \log(\lambda/N) \right]}.$$

### Genetic Algorithm

As mentioned in Section 6, while the GA has a relatively clean theory that applies in both finite and asymptotic samples, there are significant challenges in computing the elements of the Markov transition matrix  $\Pi$ . The number of possible states—corresponding to the number  $N$  of possible populations—grows extremely rapidly with the number of population elements  $N$  or the number of bits  $b$ . The computation of the  $N_{\text{pop}} \times N_{\text{pop}}$  transition matrix  $\Pi$  quickly overwhelms even the most powerful current personal computers.

Nevertheless, in principle, the Markov structure is convenient for establishing a convergence rate for the GA. The full version of the paper provides value for  $n$ .

### 7.2 Application of Convergence Rate Expressions for Varying $p$

We now apply the results above to demonstrate relative efficiency for varying  $p$ . Because the GA result is computationally explosive as  $p$  gets larger (requiring a larger bit string length and/or population size), we restrict the comparison here to the four algorithms: random search, SPSA, SAN and ES. Let  $\Theta = [0, 1]^p$  (the  $p$ -dimensional hypercube with minimum and maximum  $\theta$  values of 0 and 1 for each component). We want to guarantee with probability 0.90 that each element of  $\theta$  is within 0.04 units of the optimal. Let the (unknown) true  $\theta, \theta^*$ , lie in  $(0.04, 0.96)^p$ . The individual components of  $\theta^*$  are  $\theta_i^*$ . Hence,

$$S(\theta^*) = [\theta_1^* - 0.04, \theta_1^* + 0.04] \times [\theta_2^* - 0.04, \theta_2^* + 0.04] \times \dots \\ \times [\theta_p^* - 0.04, \theta_p^* + 0.04] \subset \Theta.$$

Table 7.1 is a summary of relative efficiency for the setting above for  $p = 2, 5$ , and  $10$ ; the efficiency was normalized so that all algorithms performed equally at  $p = 1$ , as described below. The numbers in Table 7.1 are the ratios of the number of loss measurements for the given algorithm over the number for the best algorithm at the specified  $p$ ; the highlighted values 1.0 indicate the best algorithm for each of the values of  $p$ . To establish a fair basis for comparison, we fixed the various parameters in the expressions above (e.g.,  $\sigma$  in SPSA and SAN,  $\rho$  for the ES, etc.) so that the algorithms produced identical efficiency results for  $p = 1$  (requiring  $n = 28$  measurements to achieve the objective outlined above). These parameters do not explicitly depend on  $p$ . We then use these parameter settings as  $p$  increases.

**Table 7.1.** Ratios of loss measurements needed relative to best algorithm at each  $p$  for  $1 \leq p \leq 10$

	$p = 1$	$p = 2$	$p = 5$	$p = 10$
<i>Rand. Search</i>	1.0	11.6	8970	$2.0 \times 10^9$
<i>SPSA</i>	1.0	1.5	1.0	1.0
<i>SAN</i>	1.0	1.0	2.2	4.1
<i>ES</i>	1.0	1.9	1.9	2.8

Table 7.1 illustrates the explosive growth in the relative (and absolute) number of loss evaluations needed as  $p$  increases for the random search algorithm. The other algorithms perform more comparably, but there are still some non-negligible differences. For example, at  $p = 5$ , SAN will take 2.2 times more loss measurements than SPSA to achieve the objective of having  $\hat{\theta}_k$  inside  $S(\theta^*)$  with probability 0.90. Of course, as  $p$  increases, all algorithms take more measurements; the table only shows relative numbers of function evaluations (considered more reliable than absolute numbers).

This large improvement of SPSA and SAN relative to random search may partly result from the more restrictive regularity conditions of SPSA and SAN (i.e., for formal convergence, SPSA assumes a several-times-differentiable loss function) and partly from the fact that SPSA and SAN work with *implicit* gradient information via gradient approximations. The performance for ES is quite good. The restriction to strongly convex fitness functions, however, gives the ES in this setting a strong structure not available to the other algorithms. It remains unclear what practical theoretical conclusions can be drawn on a broader class of problems.

## REFERENCES

Arnold, D. V. (2002), *Noisy Optimization with Evolution Strategies*, Kluwer, Boston.

- Beyer, H.-G. (1995), "Toward a Theory of Evolution Strategies: On the Benefits of Sex—the  $(\mu/\mu, \lambda)$  Theory," *Evolutionary Computation*, vol. 3, pp. 81–111.
- Chin, D. C. (1997), "Comparative Study of Stochastic Algorithms for System Optimization Based On Gradient Approximations," *IEEE Transactions on Systems, Man, and Cybernetics—B*, vol. 27, pp. 244–249.
- Dippon, J. and Renz, J. (1997), "Weighted Means in Stochastic Approximation of Minima," *SIAM Journal on Control and Optimization*, vol. 35, pp. 1811–1827.
- Gelfand, S. and Mitter, S.K. (1993), "Metropolis-Type Annealing Algorithms for Global Optimization in  $R^d$ ," *SIAM Journal of Control and Optimization*, vol. 31, pp. 111–131.
- Gerencsér, L. (1999), "Convergence Rate of Moments in Stochastic Approximation with Simultaneous Perturbation Gradient Approximation and Resetting," *IEEE Transactions on Automatic Control*, vol. 44, pp. 894–905.
- Gerencsér, L. and Vágó, Z. (2000), "SPSA in Noise-Free Optimization," in *Proceedings of the American Control Conference*, pp. 3284–3288.
- Iosifescu, M. (1980), *Finite Markov Processes and Their Applications*, Wiley, New York.
- Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983), "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671–680.
- Lehmann, E.L. (1983), *Theory of Point Estimation*, Wiley, New York.
- Maryak, J.L. and Chin, D.C. (2001), "Global Random Optimization by Simultaneous Perturbation Stochastic Approximation," in *Proceedings of the American Control Conference*, pp. 756–762.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M. Teller, A. and Teller, E. (1953), "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092.
- Qi, X. and Palmeiri, F. (1994), "Theoretical Analysis of Evolutionary Algorithms with Infinite Population Size in Continuous Space, Part I: Basic Properties," *IEEE Transactions on Neural Networks*, vol. 5, pp. 102–119.
- Rappl, G. (1989), "On Linear Convergence of a Class of Random Search Algorithms," *Zeitschrift für angewandte Mathematik und Mechanik (ZAMM)*, vol. 69, pp. 37–45.
- Rudolph, G. (1997a), *Convergence Properties of Evolutionary Algorithms*, Kovac, Hamburg.
- Rudolph, G. (1997b), "Convergence Rates of Evolutionary Algorithms for a Class of Convex Objective Functions," *Control and Cybernetics*, vol. 26, pp. 375–390.

- Rudolph, G. (1998), "Finite Markov Chain Results in Evolutionary Computation: A Tour d'Horizon," *Fundamenta Informaticae*, vol. 34, pp. 1–22.
- Serfling, R.J. (1980), *Approximation Theorems of Mathematical Statistics*, Wiley, New York.
- Spall, J. C. (1992), "Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation," *IEEE Transactions on Automatic Control*, vol. 37, pp. 332–341.
- Spall, J.C. (2000), "Adaptive Stochastic Approximation by the Simultaneous Perturbation Method," *IEEE Transactions on Automatic Control*, vol. 45, pp. 1839–1853.
- Spall, J. C. (2003), *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*, Wiley, Hoboken, NJ.
- Stark, D. R. and Spall, J. C. (2001), "Computable Bounds on the Rate of Convergence in Evolutionary Computation," in *Proceedings of the American Control Conference*, pp. 918–922.
- Suzuki, J. (1995), "A Markov Chain Analysis on Simple Genetic Algorithms," *IEEE Transactions on Systems, Man, and Cybernetics—B*, vol. 25, pp. 655–659.
- Wolpert, D. H. and Macready, W. G. (1997), "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82.
- Yin, G. G. (1999), "Rates of Convergence for a Class of Global Stochastic Optimization Algorithms," *SIAM Journal on Optimization*, vol. 10, pp. 99–120.